C 13

```
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSSSSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMMMMM   MMMMMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRR         RRR  MMM   MMM   MMM  SSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM         MMM     SSSSSSSSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR   RRR        MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR      RRR     MMM         MMM              SSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSSS
RRR         RRR  MMM         MMM  SSSSSSSSSSSSS
```

```
**FILE**ID**NTOSCNXAB                              C  6




NN      NN  TTTTTTTTTT    000000      SSSSSSSS    CCCCCCCC  NN      NN  XX      XX    AAAAAA    BBBBBBBB
NN      NN  TTTTTTTTTT    000000      SSSSSSSS    CCCCCCCC  NN      NN  XX      XX    AAAAAA    BBBBBBBB
NN      NN      TT      00      00  SS          CC          NN      NN  XX      XX  AA      AA  BB      BB
NN      NN      TT      00      00  SS          CC          NN      NN  XX      XX  AA      AA  BB      BB
NNNN    NN      TT      00    0000  SS          CC          NNNN    NN    XX  XX    AA      AA  BB      BB
NNNN    NN      TT      00    0000  SS          CC          NNNN    NN    XX  XX    AA      AA  BB      BB
NN  NN  NN      TT      00  00  00    SSSSSS    CC          NN  NN  NN      XX      AA      AA  BBBBBBBB
NN  NN  NN      TT      00  00  00    SSSSSS    CC          NN  NN  NN      XX      AA      AA  BBBBBBBB
NN    NNNN      TT      0000    00          SS  CC          NN    NNNN    XX  XX    AAAAAAAAAA  BB      BB
NN    NNNN      TT      0000    00          SS  CC          NN    NNNN    XX  XX    AAAAAAAAAA  BB      BB
NN      NN      TT      00      00          SS  CC          NN      NN  XX      XX  AA      AA  BB      BB      ....
NN      NN      TT      00      00          SS  CC          NN      NN  XX      XX  AA      AA  BB      BB      ....
NN      NN      TT        000000    SSSSSSSS      CCCCCCCC  NN      NN  XX      XX  AA      AA  BBBBBBBB        ....
NN      NN      TT        000000    SSSSSSSS      CCCCCCCC  NN      NN  XX      XX  AA      AA  BBBBBBBB        ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000          $BEGIN  NTOSCNXAB,000,NF$NETWORK,<SCAN XAB CHAIN>
0000
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000    ;
0000
0000    ;++
0000    ; Facility: RMS
0000    ;
0000    ; Abstract:
0000    ;
0000    ;       This module contains routines that scan:
0000    ;       (1) the user XAB chain and examine FAL's capabilities to determine
0000    ;           which DAP Extended Attributes messages should be requested to be
0000    ;           returned by the remote FAL.
0000    ;       (2) the user Name Block and examine FAL's capabilities to determine
0000    ;           if a DAP (resultant) Name message should be requested to be
0000    ;           returned by the remote FAL.
0000    ;
0000    ; Environment: VAX/VMS, executive mode
0000    ;
0000    ; Author: James A. Krycka,      Creation Date:  05-JUN-1979
0000    ;
0000    ; Modified By:
0000    ;
0000    ;       V03-006 JAK0119         J A Krycka      16-JUL-1983
0000    ;               Scan the Journaling XAB and save the JOP field for use by
0000    ;               NT$CREATE.
0000    ;
0000    ;       V03-005 JAK0115         J A Krycka      29-JUN-1983
0000    ;               Support probe of extended Protection XAB.
0000    ;
0000    ;       V03-004 KRM0110         K Malik         23-May-1983
0000    ;               Update to support DAP V7.0 spec.
0000    ;
```

```
                                   0000     58 ;        V03-003 KPL0001         Peter Lieberwirth         23-May-1983
                                   0000     59 ;                Fix branch destinations that are out of range.
                                   0000     60 ;
                                   0000     61 ;        V03-002 KRM0050         K R Malik         02-Jun-1982
                                   0000     62 ;                Fix minor bug in NT$SCAN_NAMBLK.
                                   0000     63 ;
                                   0000     64 ;--
```

```
NT
Sy

NW
NW
NW
NW
NW
NW
NW
NW
NW
NW
NW
NW
NW
NW
NW
NW
NW
NX
PR
RE
RM
RM
RM
RM
SE
SU
VA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA
XA



PS
--
.
NF
$A
```

```
0000           .SBTTL  DECLARATIONS
0000
0000   ;
0000   ; Include Files:
0000   ;
0000
0000           $DAPCNFDEF                              ; Define DAP Configuration message
0000           $DAPACCDEF                              ; Define DAP Access message
0000           $FABDEF                                 ; Define File Access Block symbols
0000           $IFBDEF                                 ; Define IFAB symbols
0000           $NAMDEF                                 ; Define Name Block symbols
0000           $NWADEF                                 ; Define Network Work Area symbols
0000           $XABDEF                                 ; Define symbols common to all XABs
0000           $XABALLDEF                              ; Define Allocation XAB symbols
0000           $XABDATDEF                              ; Define Date and Time XAB symbols
0000           $XABFHCDEF                              ; Define File Header Char symbols
0000           $XABKEYDEF                              ; Define Key Definition XAB symbols
0000           $XABPRODEF                              ; Define Protection XAB symbols
0000           $XABRDTDEF                              ; Define Revision Date/Time XAB symbols
0000           $XABSUMDEF                              ; Define Summary XAB symobls
0000   ;       $XABTRMDEF                              ; Define Terminal XAB symbols
0000   ;       $XABCXFDEF                              ; Define FAB Context XAB symbols
0000   ;       $XABCXRDEF                              ; Define RAB Context XAB symbols
0000           $XABJNLDEF                              ; Define Journal XAB symbols
0000
0000   ;
0000   ; Macros:
0000   ;
0000   ;       None
0000   ;
0000   ; Equated Symbols:
0000   ;
0000
0000           ASSUME  NWA$Q_FLG EQ 0
0000
0000   ;
0000   ; Own Storage:
0000   ;
0000   ;       None
0000   ;
```

```
0000            .SBTTL  NT$SCAN_XABCHN - SCAN XAB CHAIN
0000
0000    ;++
0000    ; NT$SCAN_XABCHN - scans the user XAB chain and examines FAL's capabilities
0000    ;       to determine which DAP Attributes and Extended Attributes messages to
0000    ;       request the remote FAL to return. It also verifies that all Allocation
0000    ;       XABs found are chained sequentially and that all Key Definition XABs
0000    ;       found are chained sequentially (i.e., they form sub-chains).
0000    ;
0000    ;       The message request mask is returned in R2.
0000    ;
0000    ; Calling Sequence:
0000    ;
0000    ;       BSBW    NT$SCAN_XABCHN
0000    ;
0000    ; Input Parameters:
0000    ;
0000    ;       R6      Close operation flag
0000    ;       R7      NWA (=DAP) address
0000    ;       R8      FAB address
0000    ;       R9      IFAB address
0000    ;       R10     IFAB/FWA address
0000    ;       R11     Impure Area address
0000    ;
0000    ; Implicit Inputs:
0000    ;
0000    ;       User ALL, DAT, FHC, JNL, KEY, PRO, RDT, and SUM XABs
0000    ;       DAP$Q_SYSCAP bits KEYXAB, ALLXAB, SUMXAB, TIMXAB, PROXAB,
0000    ;                         CHGTIMCLS, CHGPROCLS
0000    ;
0000    ; Output Parameters:
0000    ;
0000    ;       R0      Status code (RMS)
0000    ;       R1      Destroyed
0000    ;       R2      Message request mask
0000    ;       R3-R5   Destroyed
0000    ;
0000    ; Implicit Outputs:
0000    ;
0000    ;       NWA$B_ALLXABCNT
0000    ;       NWA$B_KEYXABCNT
0000    ;       NWA$W_JNLXABJOP
0000    ;       NWA$L_ALLXABADR
0000    ;       NWA$L_DATXABADR
0000    ;       NWA$L_FHCXABADR
0000    ;       NWA$L_KEYXABADR
0000    ;       NWA$L_PROXABADR
0000    ;       NWA$L_RDTXABADR
0000    ;       NWA$L_SUMXABADR
0000    ;
0000    ; Completion Codes:
0000    ;
0000    ;       Standard RMS completion codes
0000    ;
0000    ; Side Effects:
0000    ;
0000    ;       User XABs are probed for writeability.
```

```
                    0000  ;
                    0000  ;--
                    0000
                    0000  NT$SCAN_XABCHN::                              ; Entry point
                    0000          $ZERO_FILL-                           ; Zero XAB scan results block
                    0000                  DST=NWA$T_SCAN(R7)-           ;  Address of block
                    0000                  SIZE=#NWA$C_SCAN              ;  Length of block
                    000A                                                ; Zero R2 (request mask) as side effect
                    000A                                                ;  of executing a MOVC5 instruction
53    24 AB    D0   000A          MOVL    FAB$L_XAB(R8),R3              ; Get first XAB address in chain
         07    11   000E          BRB     CHKXAB                        ;
                    0010
                    0010  ;+
                    0010  ; Process next XAB in the chain.
                    0010  ;
                    0010  ; Note: XABs not supported by DECnet (if found in the XAB chain) will be
                    0010  ;       ignored.
                    0010  ;-
                    0010
                    0010          ASSUME  XAB$C_DAT EQ 18
                    0010          ASSUME  XAB$C_PRO EQ 19
                    0010          ASSUME  XAB$C_ALL EQ 20
                    0010          ASSUME  XAB$C_KEY EQ 21
                    0010          ASSUME  XAB$C_SUM EQ 22
                    0010          ASSUME  XAB$C_FHC EQ 29
                    0010          ASSUME  XAB$C_RDT EQ 30
                    0010  ;       ASSUME  XAB$C_TRM EQ 31               ; Not supported for network use
                    0010  ;       ASSUME  XAB$C_CXF EQ 32               ; Not supported for network use
                    0010  ;       ASSUME  XAB$C_CXR EQ 33               ; Not supported for network use
                    0010          ASSUME  XAB$C_JNL EQ 34               ; Not supported for network use
                    0010                                                ;  BUT will be processed to obtain
                    0010                                                ;  JOP field for use by NT$CREATE
                    0010
   54    53    D0   0010  NXTXAB: MOVL    R3,R4                         ; Save address of current XAB in chain
53    04 A3    D0   0013          MOVL    XAB$L_NXT(R3),R3              ; Get address of next XAB in chain
         30    13   0017  CHKXAB: BEQL    EXIT                          ; Branch if none
                    0019          IFNORD  #XAB$L_NXT+4,(R3),-           ; Probe for readability thru NXT field
                    0019                  ERRXAB,IFB$B_MODE(R9)         ;  of XAB and branch on failure
      ED AF    9F   0020          PUSHAB  B^NXTXAB                      ; Push return address on stack
                    0023          $CASEB  SELECTOR=XAB$B_COD(R3)-       ; Dispatch to routine to process:
                    0023                  BASE=#XAB$C_DAT-              ;
                    0023                  DISPL=<-                      ;
                    0023                          DATE_TIME-            ;  XABDAT
                    0023                          PROTECTION-           ;  XABPRO
                    0023                          ALLOCATION-           ;  XABALL
                    0023                          KEY_DEFINITION-       ;  XABKEY
                    0023                          SUMMARY-              ;  XABSUM
                    0023                          ERRCOD-               ;  Invalid XAB type
                    0023                          ERRCOD-               ;  Invalid XAB type
                    0023                          ERRCOD-               ;  Invalid XAB type
                    0023                          ERRCOD-               ;  Invalid XAB type
                    0023                          ERRCOD-               ;  Invalid XAB type
                    0023                          ERRCOD-               ;  Invalid XAB type
                    0023                          FILE_HEADER-          ;  XABFHC
                    0023                          REV_DATE_TIME-        ;  XABRDT
                    0023                          EXIT-                 ;  Ignore XABTRM
                    0023                          EXIT-                 ;  Ignore XABCXF
```

```
                   0023                           EXIT-                  ; Ignore XABCXR
                   0023                           JOURNALING-            ; XABJNL
                   0023                    >                             ;
                   0049
                   0049  ;+
                   0049  ; Exit paths.
                   0049  ;-
                   0049
                   0049  EXIT:    RMSSUC                                 ; Return success
               05  004C           RSB                                    ; Exit with RMS code in R0
          01   BA  004D  ERRCOD:  POPR     #^M<R0>                       ; Discard return address
                   004F           RMSERR   COD                           ; Invalid XAB type code
          10   11  0054           BRB      SETSTV                        ;
          01   BA  0056  ERRIMX:  POPR     #^M<R0>                       ; Discard return address
                   0058           RMSERR   IMX                           ; Duplicate XAB or XABs are not dense
          07   11  005D           BRB      SETSTV                        ;
          03   BA  005F  ERRXAB2:POPR      #^M<R0,R1>                    ; Discard return addresses
                   0061  ERRXAB:  RMSERR   XAB                           ; XAB too short or not accessible
 0C A8    53   D0  0066  SETSTV:  MOVL     R3,FAB$L_STV(R8)              ; Return XAB address in STV field
               05  006A           RSB                                    ; Exit with RMS code in R0
                   006B
                   006B  ;+
                   006B  ; This routine checks the control block for correct length and writeability.
                   006B  ;-
                   006B
                   006B  VALIDATE_XAB:                                  ; Entry point
 50    01 A3   9A  006B           MOVZBL   XAB$B_BLN(R3),R0              ; Get stated length of block
    51    50   D1  006F           CMPL     R0,R1                         ; Compare against expected length
          EB   1F  0072           BLSSU    ERRXAB2                       ; Branch if too small
                   0074           IFNOWRT  R0,(R3),ERRXAB2,-             ; Probe for writeability and branch on
                   0074                    IFB$B_MODE(R9)                ;  failure
               05  007B           RSB                                    ; Exit
                   007C
                   007C  ;+
                   007C  ; This routine handles the Date and Time XAB.
                   007C  ;-
                   007C
                   007C  DATE_TIME:                                     ; Entry point
    51    24   9A  007C           MOVZBL   #XAB$C_DATLEN_V2,R1           ; Get minimum (i.e., V2) length of XAB
          EA   10  007F           BSBB     VALIDATE_XAB                  ; Check length and accessibility
  0104    C7   D5  0081           TSTL     NWA$L_DATXABADR(R7)           ; Declare error as this is a duplicate
          CF   12  0085           BNEQ     ERRIMX                        ;  XAB
0104 C7   53   D0  0087           MOVL     R3,NWA$L_DATXABADR(R7)        ; Save address of Date and Time XAB
          1A   E1  008C           BBC      #DAP$V_TIMXAB,-               ; Branch if Date and Time message is
   07 28  A7       008E                    DAP$Q_SYSCAP(R7),10$          ;  not supported by partner
      04  56   E8  0091           BLBS     R6,10$                        ; This XAB is not an input on close
                   0094           $SETBIT  #DAP$V_DSP_TIM,R2             ; Update request mask
               05  0098  10$:     RSB                                    ; Exit
                   0099
                   0099  ;+
                   0099  ; This routine handles the Protection XAB.
                   0099  ;-
                   0099
                   0099  PROTECTION:                                    ; Entry point
    51    10   9A  0099           MOVZBL   #XAB$C_PROLEN_V3,R1           ; Get minimum (i.e., V3) length of XAB
          CD   10  009C           BSBB     VALIDATE_XAB                  ; Check length and accessibility
  0110    C7   D5  009E           TSTL     NWA$L_PROXABADR(R7)           ; Declare error as this is a duplicate
```

```
             B2   12 00A2               BNEQ    ERRIMX                      ;  XAB
     0110 C7   53   D0 00A4               MOVL    R3,NWA$L_PROXABADR(R7)      ; Save address of Protection XAB
             1B   E1 00A9               BBC     #DAP$V_PROXAB,-             ; Branch if Protection message is
       0C 28 A7      00AB                       DAP$Q_SYSCAP(R7),20$        ;  not supported by partner
          05 56   E9 00AE               BLBC    R6,10$                      ; An additional system capabilities
             2C   E1 00B1               BBC     #DAP$V_CHGPROCLS,-          ;  check is required if this is a
       04 28 A7      00B3                       DAP$Q_SYSCAP(R7),20$        ;  change operation
                     00B6       10$:    $SETBIT #DAP$V_DSP_PRO,R2           ; Update request mask
                  05 00BA       20$:    RSB                                 ; Exit
                     00BB
                     00BB       ;+
                     00BB       ; This routine handles the Allocation XAB.
                     00BB       ;-
                     00BB
                     00BB       ALLOCATION:                                 ; Entry point
        51    20   9A 00BB              MOVZBL  #XAB$C_ALLLEN,R1            ; Get length of XAB
              AB   10 00BE              BSBB    VALIDATE_XAB                ; Check length and accessibility
         011C C7   96 00C0              INCB    NWA$B_ALLXABCNT(R7)         ; Increment XAB counter
  01     011C C7   91 00C4              CMPB    NWA$B_ALLXABCNT(R7),#1      ; Branch if this is first
              08   13 00C9              BEQL    10$                         ;  Allocation XAB in chain
     14       64   91 00CB              CMPB    XAB$B_COD(R4),#XAB$C_ALL; Check previous XAB in chain;
              11   13 00CE              BEQL    20$                         ;  it must also be an Allocation XAB
            FF83   31 00D0              BRW     ERRIMX                      ;  else this XAB is out of order
     0100 C7  53   D0 00D3      10$:    MOVL    R3,NWA$L_ALLXABADR(R7)      ; Save address of first Allocation XAB
              17   E1 00D8              BBC     #DAP$V_ALLXAB,-             ; Branch if Allocation message is
       04 28 A7       00DA                      DAP$Q_SYSCAP(R7),20$        ;  not supported by partner
                      00DD              $SETBIT #DAP$V_DSP_ALL,R2           ; Update request mask
                   05 00E1      20$:    RSB                                 ; Exit
                      00E2
                      00E2      ;+
                      00E2      ; This routine handles the Key Definition XAB.
                      00E2      ;-
                      00E2
                      00E2      KEY_DEFINITION:                             ; Entry point
   51      40 8F   9A 00E2              MOVZBL  #XAB$C_KEYLEN_V2,R1         ; Get minimum (i.e., V2) length of XAB
            FF82   30 00E6              BSBW    VALIDATE_XAB                ; Check length and accessibility
         011D C7   96 00E9              INCB    NWA$B_KEYXABCNT(R7)         ; Increment XAB counter
  01     011D C7   91 00ED              CMPB    NWA$B_KEYXABCNT(R7),#1      ; Branch if this is first
              08   13 00F2              BEQL    10$                         ;  Key Definition XAB in chain
     15       64   91 00F4              CMPB    XAB$B_COD(R4),#XAB$C_KEY; Check previous XAB in chain;
              11   13 00F7              BEQL    20$                         ;  it must also be a Key Definition XAB
            FF5A   31 00F9              BRW     ERRIMX                      ;  else this XAB is out of order
     010C C7  53   D0 00FC      10$:    MOVL    R3,NWA$L_KEYXABADR(R7)      ; Save address of first Key Def XAB
              16   E1 0101              BBC     #DAP$V_KEYXAB,-             ; Branch if Key Definition message is
       04 28 A7       0103                      DAP$Q_SYSCAP(R7),20$        ;  not supported by partner
                      0106              $SETBIT #DAP$V_DSP_KEY,R2           ; Update request mask
                   05 010A      20$:    RSB                                 ; Exit
                      010B
                      010B      ;+
                      010B      ; This routine handles the Summary XAB.
                      010B      ;-
                      010B
                      010B      SUMMARY:                                    ; Entry point
        51    0C   9A 010B              MOVZBL  #XAB$C_SUMLEN,R1            ; Get length of XAB
            FF5A   30 010E              BSBW    VALIDATE_XAB                ; Check length and accessibility
         0118 C7   D5 0111              TSTL    NWA$L_SUMXABADR(R7)         ; Declare error as this is a duplicate
              55   12 0115              BNEQ    ERRIMX1                     ;  XAB
  0118 C7   53   D0 0117               MOVL    R3,NWA$L_SUMXABADR(R7)  ; Save address of Summary XAB
            18   E1 011C               BBC     #DAP$V_SUMXAB,-         ; Branch if Summary message is
     04 28 A7       011E                       DAP$Q_SYSCAP(R7),10$    ;  not supported by partner
                    0121               $SETBIT #DAP$V_DSP_SUM,R2       ; Update request mask
                05  0125       10$:    RSB                             ; Exit
                    0126
                    0126       ;+
                    0126       ; This routine handles the File Header Characteristics XAB.
                    0126       ;
                    0126       ; Note: The File Header Characteristics XAB is supported in DAP through the
                    0126       ;       DAP Attributes message. Thus there is no system capabilities check
                    0126       ;       associated with it.
                    0126       ;-
                    0126
                    0126       FILE_HEADER:                            ; Entry point
       51   2C   9A 0126               MOVZBL  #XAB$C_FHCLEN,R1        ; Get length of XAB
          FF3F   30 0129               BSBW    VALIDATE_XAB            ; Check length and accessibility
     0108 C7     D5 012C               TSTL    NWA$L_FHCXABADR(R7)     ; Declare error as this is a duplicate
            3A   12 0130               BNEQ    ERRIMX1                 ;  XAB
  0108 C7   53   D0 0132               MOVL    R3,NWA$L_FHCXABADR(R7)  ; Save address of File Header Char XAB
                    0137               $SETBIT #DAP$V_DSP_ATT,R2       ; Update request mask
                05  013B               RSB                             ; Exit
                    013C
                    013C       ;+
                    013C       ; This routine handles the Revision Date and Time XAB.
                    013C       ;
                    013C       ; Note: Both the Date and Time XAB and the Revision Date and Time XAB are
                    013C       ;       supported in DAP through the DAP Date and Time message.
                    013C       ;-
                    013C
                    013C       REV_DATE_TIME:                          ; Entry point
       51   14   9A 013C               MOVZBL  #XAB$C_RDTLEN,R1        ; Get length of XAB
          FF29   30 013F               BSBW    VALIDATE_XAB            ; Check length and accessibility
     0114 C7     D5 0142               TSTL    NWA$L_RDTXABADR(R7)     ; Declare error as this is a duplicate
            24   12 0146               BNEQ    ERRIMX1                 ;  XAB
  0114 C7   53   D0 0148               MOVL    R3,NWA$L_RDTXABADR(R7)  ; Save address of Rev Date and Time XAB
            1A   E1 014D               BBC     #DAP$V_TIMXAB,-         ; Branch if Date and Time message
     0C 28 A7       014F                       DAP$Q_SYSCAP(R7),20$    ;  not supported by partner
        05  56   E9 0152               BLBC    R6,10$                  ; An additional system capabilities
            2B   E1 0155               BBC     #DAP$V_CHGTIMCLS,-      ;  check is required if this is a
     04 28 A7       0157                       DAP$Q_SYSCAP(R7),20$    ;  change operation
                    015A       10$:    $SETBIT #DAP$V_DSP_TIM,R2       ; Update request mask
                05  015E       20$:    RSB                             ; Exit
                    015F
                    015F       ;+
                    015F       ; This routine handles the Journaling XAB.
                    015F       ;
                    015F       ; Note: This XAB is not supported for network use--it will be ignored unless
                    015F       ;       the journaling options field is non-zero on create. Consequently, this
                    015F       ;       routine saves the JOP field in the NWA for use by NT$CREATE.
                    015F       ;-
                    015F
                    015F       JOURNALING:                             ; Entry point
       51   3C   9A 015F               MOVZBL  #XAB$C_JNLLEN,R1        ; Get length of XAB
          FF06   30 0162               BSBW    VALIDATE_XAB            ; Check length and accessibility
        08 A3    B0 0165               MOVW    XAB$W_JOP(R3),-         ; Save journaling options value in NWA
     011E C7        0168                       NWA$W_JNLXABJOP(R7)     ;
```

```
                              05  016B    392            RSB                                ; Exit
                                  016C    393
                        FEE7  31  016C    394 ERRIMX1:BRW      ERRIMX                       ; Branch aid
```

```
                 016F                .SBTTL  NT$SCAN_KEYXAB - SCAN KEY DEFINITION XAB
                 016F                .SBTTL  NT$SCAN_ALLXAB - SCAN ALLOCATION XAB
                 016F      ;++
                 016F      ; NT$SCAN_KEYXAB - scans a specific Key Definition XAB without scanning the
                 016F      ;         entire XAB chain.
                 016F      ; NT$SCAN_ALLXAB - scans a specific Allocation XAB without scanning the
                 016F      ;         entire XAB chain.
                 016F      ;
                 016F      ; Calling Sequence:
                 016F      ;
                 016F      ;         BSBW    NT$SCAN_KEYXAB
                 016F      ;         BSBW    NT$SCAN_ALLXAB
                 016F      ;
                 016F      ; Input Parameters:
                 016F      ;
                 016F      ;         R6      Allocation or Key Definition XAB address
                 016F      ;         R7      NWA (=DAP) address
                 016F      ;         R8      FAB address
                 016F      ;         R9      IFAB address
                 016F      ;         R10     IFAB/FWA address
                 016F      ;         R11     Impure Area address
                 016F      ;
                 016F      ; Implicit Inputs:
                 016F      ;
                 016F      ;         None
                 016F      ;
                 016F      ; Output Parameters:
                 016F      ;
                 016F      ;         R0      Status code (RMS)
                 016F      ;         R1      Destroyed
                 016F      ;         R3      Destroyed
                 016F      ;
                 016F      ; Implicit Outputs:
                 016F      ;
                 016F      ;         None
                 016F      ;
                 016F      ; Completion Codes:
                 016F      ;
                 016F      ;         Standard RMS Completion codes
                 016F      ;
                 016F      ; Side Effects:
                 016F      ;
                 016F      ;         User XAB is probed for writeability
                 016F      ;
                 016F      ;--
                 016F
                 016F      NT$SCAN_KEYXAB::                                  ; Entry point
51    40 8F   9A 016F              MOVZBL  #XAB$C_KEYLEN_V2,R1         ; Get minimum (i.e., V2) length of XAB
         03   11 0173              BRB     COMMON_SCAN                 ; Join common code
                 0175      NT$SCAN_ALLXAB::                                  ; Entry point
   51    20   9A 0175              MOVZBL  #XAB$C_ALLLEN,R1            ; Get length of XAB
                 0178      COMMON_SCAN:                                      ; Common code
   53    56   D0 0178              MOVL    R6,R3                       ; Get address of XAB to probe
                 017B              IFNORD  #XAB$L_NXT+4,(R3),-         ; Probe for readability thru NXT field
                 017B                      10$,IFB$B_MODE(R9)          ;  of XAB and branch on failure
50    01 A3   9A 0182              MOVZBL  XAB$B_BLN(R3),R0            ; Get stated length of block
   51    50   D1 0186              CMPL    R0,R1                       ; Compare against expected length
```

```
     0B   1F  0189   453        BLSSU   10$                          ; Branch if too small
              018B   454        IFNOWRT R0,(R3),10$,-                ; Probe for writeability and branch on
              018B   455                IFB$B_MODE(R9)               ;  failure
              0192   456        RMSSUC                               ; Return success
          05  0195   457        RSB                                  ; Exit
   FEC8   31  0196   458 10$:   BRW     ERRXAB                       ; Failure
```

```
                         0199          .SBTTL  NT$SCAN_NAMBLK - SCAN NAME BLOCK
                         0199
                         0199     ;++
                         0199     ; NT$SCAN_NAMBLK - scans the user Name Block and checks FAL's capabilities
                         0199     ;       to determine if a DAP (resultant) Name message should be requested
                         0199     ;       to be returned by the remote FAL.
                         0199     ;
                         0199     ;       An updated message request mask is returned in R2.
                         0199     ;
                         0199     ; Calling Sequence:
                         0199     ;
                         0199     ;       BSBW    NT$SCAN_NAMBLK
                         0199     ;
                         0199     ; Input Parameters:
                         0199     ;
                         0199     ;       R2      Message request mask
                         0199     ;       R7      NWA (=DAP) address
                         0199     ;       R8      FAB address
                         0199     ;       R9      IFAB address
                         0199     ;       R10     IFAB/FWA address
                         0199     ;       R11     Impure Area address
                         0199     ;
                         0199     ; Implicit Inputs:
                         0199     ;
                         0199     ;       User Name Block
                         0199     ;
                         0199     ; Output Parameters:
                         0199     ;
                         0199     ;       R0      Status code (RMS)
                         0199     ;       R1      Destroyed
                         0199     ;       R2      Updated message request mask
                         0199     ;
                         0199     ; Implicit Outputs:
                         0199     ;
                         0199     ;       None
                         0199     ;
                         0199     ; Completion Codes:
                         0199     ;
                         0199     ;       Standard RMS Completion codes
                         0199     ;
                         0199     ; Side Effects:
                         0199     ;
                         0199     ;       User Name Block is probed for writeability
                         0199     ;
                         0199     ;--
                         0199
                         0199     NT$SCAN_NAMBLK::                              ; Entry point
       00C0 8F     BB    0199             PUSHR   #^M<R6,R7>                   ; Save registers used
  57     28 A8     D0    019D             MOVL    FAB$L_NAM(R8),R7             ; Get Name block address
            19     13    01A1             BEQL    10$                          ; Branch if none
         FE5A'     30    01A3             BSBW    RM$CHKNAM                    ; Check Name block validity
         13 50     E9    01A6             BLBC    R0,10$                       ; Branch on error
         04 A7     D5    01A9             TSTL    NAM$L_RSA(R7)                ; Check for resultant string address
            0E     13    01AC             BEQL    10$                          ; Branch if none
       00C0 8F     BA    01AE             POPR    #^M<R6,R7>                   ; Restore registers
            28     E1    01B2             BBC     #DAP$V_NAMMSG,-              ; Branch if partner does not support
      09 28 A7           01B4                     DAP$Q_SYSCAP(R7),20$         ;  Name message
```

```
                                  01B7     517           $SETBIT #DAP$V_DSP_NAM,R2             ; Request Name message
                              05  01B8     518           RSB                                   ; Exit
                   00C0 8F    BA  01BC     519 10$:      POPR    #^M<R6,R7>                    ; Restore registers
                              05  01C0     520 20$:      RSB                                   ; Exit
                                  01C1     521
                                  01C1     522           .END                                  ; End of module
```

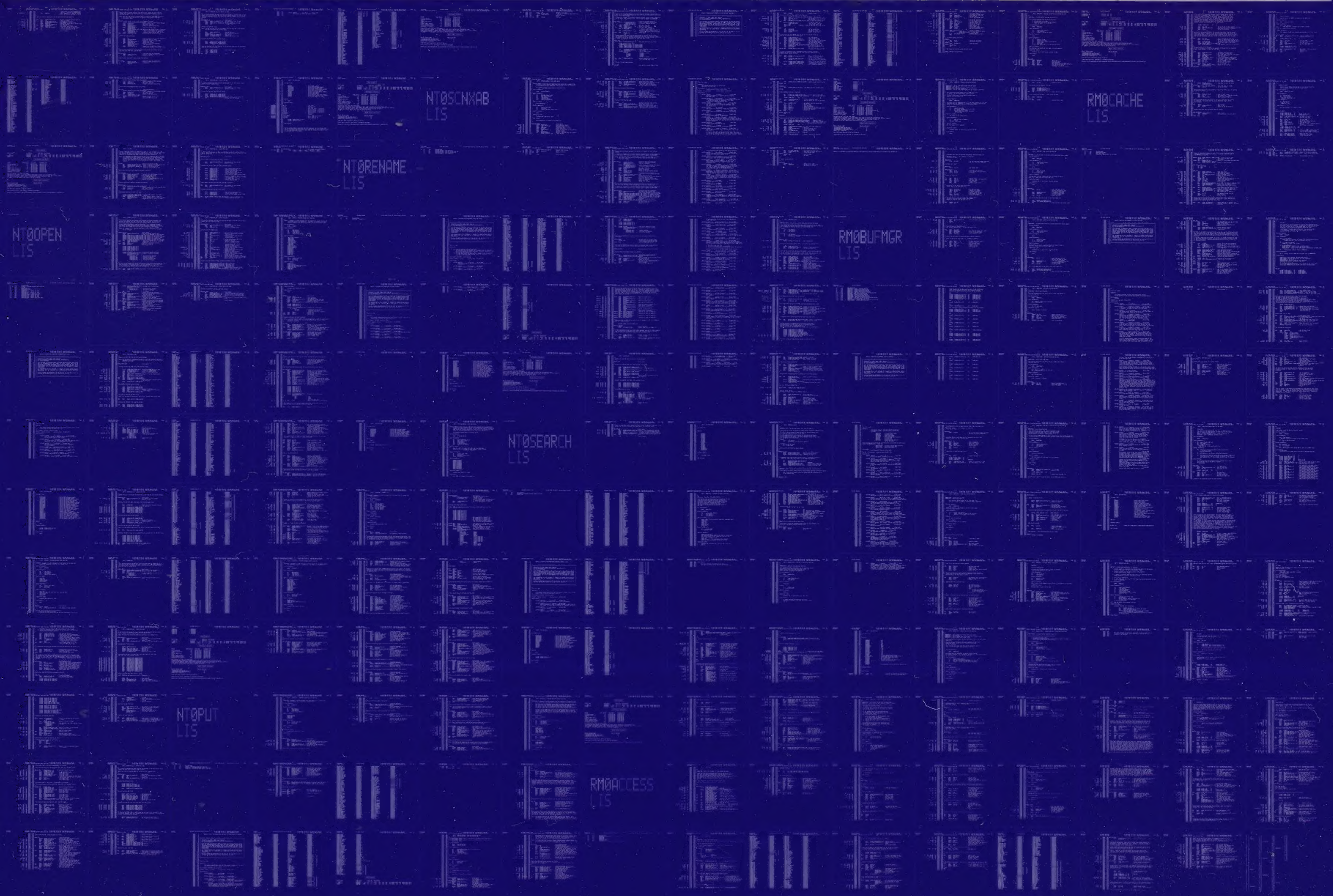
```
$$.PSECT_EP                  = 00000000                 IFB$B_MODE                   = 0000000A
$$COUNT                      = 00000011                 JOURNALING                     0000015F R       01
$$RMSTEST                    = 0000001A                 KEY_DEFINITION                 000000E2 R       01
$$RMS_PBUGCHK                = 00000010                 NAM$L_RSA                    = 00000004
$$RMS_TBUGCHK                = 00000008                 NT$SCAN_ALLXAB                 00000175 RG      01
$$RMS_UMODE                  = 00000004                 NT$SCAN_KEYXAB                 0000016F RG      01
ALLOCATION                     000000BB R       01      NT$SCAN_NAMBLK                 00000199 RG      01
CHKXAB                         00000017 R       01      NT$SCAN_XABCHN                 00000000 RG      01
COMMON_SCAN                    00000178 R       01      NWA$B_ALLXABCNT                0000011C
DAP$B_ACCFUNC                  00000040                 NWA$B_DAP_RAC                  000000C9
DAP$B_ACCOPT                   00000041                 NWA$B_FILESYS                  000000C5
DAP$B_DECVER                   00000047                 NWA$B_KEYXABCNT                0000011D
DAP$B_ECONUM                   00000045                 NWA$B_NETSTRSIZ                0000016F
DAP$B_FAC                      00000042                 NWA$B_NODBUFSIZ                00000168
DAP$B_FILESYS                  00000043                 NWA$B_ORG                      000000C6
DAP$B_OSTYPE                   00000042                 NWA$B_OSTYPE                   000000C4
DAP$B_SHR                      00000043                 NWA$B_RFM                      000000C7
DAP$B_USRNUM                   00000046                 NWA$B_RMS_RAC                  000000C8
DAP$B_USRVER                   00000048                 NWA$C_BLN                      00000800
DAP$B_VERNUM                   00000044                 NWA$C_SCAN                   = 00000020
DAP$M_DSP_3NAM               = 00000200                 NWA$K_BLN                      00000800
DAP$M_GET                    = 00000002                 NWA$L_ALLXABADR                00000100
DAP$M_GO_NOGO                = 00000010                 NWA$L_DATXABADR                00000104
DAP$M_MSE                    = 00000010                 NWA$L_DEV                      000000C0
DAP$M_TMP1$                  = 000000C0                 NWA$L_FHCXABADR                00000108
DAP$M_TMP2$                  = 0000FC00                 NWA$L_KEYXABADR                0000010C
DAP$Q_FILESPEC                 00000044                 NWA$L_MSG_MASK                 000000D4
DAP$Q_PASSWORD                 00000050                 NWA$L_PROXABADR                00000110
DAP$Q_SYSCAP                   00000028                 NWA$L_RDTXABADR                00000114
DAP$V_ALLXAB                 = 00000017                 NWA$L_SAVE_FLGS                00000128
DAP$V_CHGPROCLS              = 0000002C                 NWA$L_SUMXABADR                00000118
DAP$V_CHGTIMCLS              = 0000002B                 NWA$L_THREAD                   000000FC
DAP$V_DSP_ALL                = 00000002                 NWA$L_XLTATTR                  00000238
DAP$V_DSP_ATT                = 00000000                 NWA$L_XLTBUFFLG                0000022C
DAP$V_DSP_KEY                = 00000001                 NWA$L_XLTCNT                   00000228
DAP$V_DSP_NAM                = 00000008                 NWA$L_XLTMAXINDX               00000234
DAP$V_DSP_PRO                = 00000005                 NWA$L_XLTSIZ                   00000230
DAP$V_DSP_SUM                = 00000003                 NWA$Q_ACS                      00000244
DAP$V_DSP_TIM                = 00000004                 NWA$Q_BIGBUF                   00000170
DAP$V_KEYXAB                 = 00000016                 NWA$Q_BLD                      000000F0
DAP$V_NAMMSG                 = 00000028                 NWA$Q_FLG                      00000000
DAP$V_PROXAB                 = 0000001B                 NWA$Q_INODE                    0000025C
DAP$V_SUMXAB                 = 00000018                 NWA$Q_IOSB                     000000D8
DAP$V_TIMXAB                 = 0000001A                 NWA$Q_LNODE                    00000160
DAP$W_BUFSIZ                   00000040                 NWA$Q_LOGNAME                  0000023C
DAP$W_DISPLAY1                 0000004C                 NWA$Q_NCB                      00000264
DATE_TIME                      0000007C R       01      NWA$Q_RCV                      000000E0
ERRCOD                         0000004D R       01      NWA$Q_SAVE_DESC                00000120
ERRIMX                         00000056 R       01      NWA$Q_XLTBUF1                  0000024C
ERRIMX1                        0000016C R       01      NWA$Q_XLTBUF2                  00000254
ERRXAB                         00000061 R       01      NWA$Q_XMT                      000000E8
ERRXAB2                        0000005F R       01      NWA$T_ACSBUF                   0000026C
EXIT                           00000049 R       01      NWA$T_AUXBUF                   000005E0
FAB$L_NAM                    = 00000028                 NWA$T_DAP                      00000000
FAB$L_STV                    = 0000000C                 NWA$T_INODEBUF                 000004AC
FAB$L_XAB                    = 00000024                 NWA$T_ITM_ATTR                 00000200
FILE_HEADER                    00000126 R       01      NWA$T_ITM_END                  00000224
```

```
NWA$T_ITM_LST                       00000200
NWA$T_ITM_MAXINDX                   00000218
NWA$T_ITM_STRING                    0000020C
NWA$T_NCBBUF                        0000052C
NWA$T_NODEBUF                       00000169
NWA$T_RCVBUF                        000001A0
NWA$T_SCAN                          00000100
NWA$T_TEMP                          00000120
NWA$T_XLTBUF1                       000002AC
NWA$T_XLTBUF2                       000003AC
NWA$T_XMTBUF                        000003C0
NWA$W_BUILD                         000000D2
NWA$W_DAPBUFSIZ                     000000CA
NWA$W_DIR_OFF                       000000CC
NWA$W_DISPLAY                       000000D0
NWA$W_FIL_OFF                       000000CE
NWA$W_JNLXABJOP                     0000011E
NXTXAB                              00000010 R       01
PROTECTION                          00000099 R       01
REV_DATE_TIME                       0000013C R       01
RMS$CHKNAM                          ********     X   01
RMS$_COD                            ********     X   01
RMS$_IMX                            ********     X   01
RMS$_XAB                            ********     X   01
SETSTV                              00000066 R       01
SUMMARY                             0000010B R       01
VALIDATE_XAB                        0000006B R       01
XAB$B_BLN                         = 00000001
XAB$B_COD                         = 00000000
XAB$C_ALL                         = 00000014
XAB$C_ALLLEN                      = 00000020
XAB$C_DAT                         = 00000012
XAB$C_DATLEN_V2                   = 00000024
XAB$C_FHC                         = 0000001D
XAB$C_FHCLEN                      = 0000002C
XAB$C_JNL                         = 00000022
XAB$C_JNLLEN                      = 0000003C
XAB$C_KEY                         = 00000015
XAB$C_KEYLEN_V2                   = 00000040
XAB$C_PRO                         = 00000013
XAB$C_PROLEN_V3                   = 00000010
XAB$C_RDT                         = 0000001E
XAB$C_RDTLEN                      = 00000014
XAB$C_SUM                         = 00000016
XAB$C_SUMLEN                      = 0000000C
XAB$L_NXT                         = 00000004
XAB$W_JOP                         = 00000008

                                                 +----------------+
                                                 ! Psect synopsis !
                                                 +----------------+

PSECT name                          Allocation          PSECT No.  Attributes
----------                          ----------          ---------  ----------
.  ABS  .                           00000000  (    0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
NF$NETWORK                          000001C1  (  449.)  01 (  1.)    PIC    USR    CON    REL    GBL NOSHR   EXE   RD   NOWRT NOVEC BYTE
$ABS$                               00000800  ( 2048.)  02 (  2.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
```

```
                                              +-------------------------+
                                              ! Performance indicators !
                                              +-------------------------+

Phase                     Page faults   CPU Time       Elapsed Time
-----                     -----------   --------       ------------
Initialization                    32    00:00:00.08    00:00:01.05
Command processing               143    00:00:00.80    00:00:04.52
Pass 1                           334    00:00:12.16    00:00:35.27
Symbol table sort                  0    00:00:01.49    00:00:03.11
Pass 2                           103    00:00:02.39    00:00:09.14
Symbol table output               20    00:00:00.15    00:00:00.38
Psect synopsis output              1    00:00:00.02    00:00:00.02
Cross-reference output             0    00:00:00.00    00:00:00.00
Assembler run totals             635    00:00:17.10    00:00:53.51

The working set limit was 1500 pages.
64416 bytes (126 pages) of virtual memory were used to buffer the intermediate code.
There were 60 pages of symbol table space allocated to hold 1101 non-local and 22 local symbols.
522 source lines were read in Pass 1, producing 14 object records in Pass 2.
34 pages of virtual memory were used to define 33 macros.

                                             +----------------------------+
                                             ! Macro library statistics !
                                             +----------------------------+

Macro library name                           Macros defined
------------------                           --------------
_$255$DUA28:[RMS.OBJ]RMS.MLB;1                          23
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                           2
_$255$DUA28:[SYSLIB]STARLET.MLB;2                        4
TOTALS (all libraries)                                  29

1326 GETS were required to define 29 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:NTOSCNXAB/OBJ=OBJ$:NTOSCNXAB MSRC$:NTOSCNXAB/UPDATE=(ENH$:NTOSCNXAB)+EXECML$/LIB+LIB$:RMS/LIB
```

0317 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY



NT0SCNXAB
LIS

RM0CACHE
LIS

NT0RENAME
LIS

NT0OPEN
LIS

RM0BUFMGR
LIS

NT0SEARCH
LIS

NT0PUT
LIS

RM0ACCESS
LIS